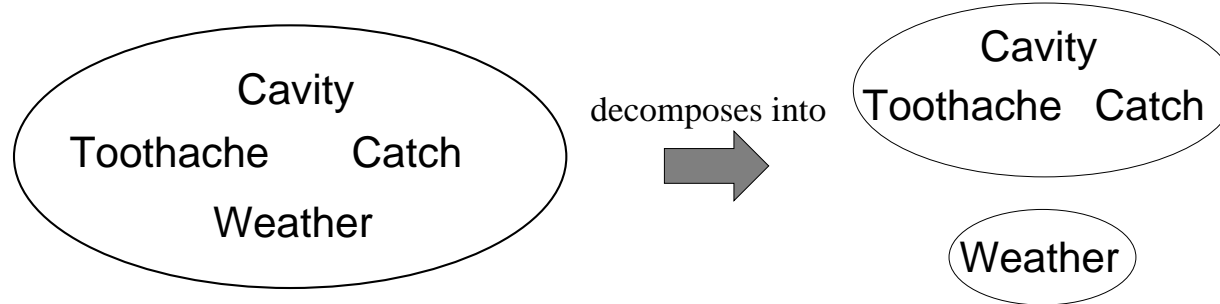


Independence

A and B are independent iff

$$\mathbf{P}(A|B) = \mathbf{P}(A) \quad \text{or} \quad \mathbf{P}(B|A) = \mathbf{P}(B) \quad \text{or} \quad \mathbf{P}(A, B) = \mathbf{P}(A)\mathbf{P}(B)$$



$$\begin{aligned} &\mathbf{P}(\textit{Toothache}, \textit{Catch}, \textit{Cavity}, \textit{Weather}) \\ &= \mathbf{P}(\textit{Toothache}, \textit{Catch}, \textit{Cavity})\mathbf{P}(\textit{Weather}) \end{aligned}$$

32 entries reduced to 12; for n independent biased coins, $2^n \rightarrow n$

Absolute independence powerful but rare

Dentistry is a large field with hundreds of variables,
none of which are independent. What to do?

Conditional independence

$\mathbf{P}(\textit{Toothache}, \textit{Cavity}, \textit{Catch})$ has $2^3 - 1 = 7$ independent entries

If I have a cavity, the probability that the probe catches in it doesn't depend on whether I have a toothache:

$$(1) \ P(\textit{catch}|\textit{toothache}, \textit{cavity}) = P(\textit{catch}|\textit{cavity})$$

The same independence holds if I haven't got a cavity:

$$(2) \ P(\textit{catch}|\textit{toothache}, \neg\textit{cavity}) = P(\textit{catch}|\neg\textit{cavity})$$

Catch is **conditionally independent** of *Toothache* given *Cavity*:

$$\mathbf{P}(\textit{Catch}|\textit{Toothache}, \textit{Cavity}) = \mathbf{P}(\textit{Catch}|\textit{Cavity})$$

Equivalent statements:

$$\mathbf{P}(\textit{Toothache}|\textit{Catch}, \textit{Cavity}) = \mathbf{P}(\textit{Toothache}|\textit{Cavity})$$

$$\mathbf{P}(\textit{Toothache}, \textit{Catch}|\textit{Cavity}) = \mathbf{P}(\textit{Toothache}|\textit{Cavity})\mathbf{P}(\textit{Catch}|\textit{Cavity})$$

Bayesian networks

A simple, graphical notation for conditional independence assertions and hence for compact specification of full joint distributions

Syntax:

- a set of nodes, one per variable

- a directed, acyclic graph (link \approx “directly influences”)

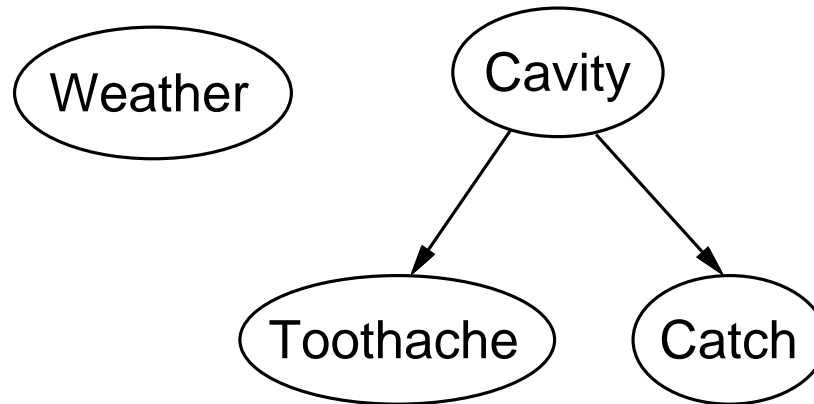
- a conditional distribution for each node given its parents:

$$\mathbf{P}(X_i | Parents(X_i))$$

In the simplest case, conditional distribution represented as a **conditional probability table** (CPT) giving the distribution over X_i for each combination of parent values

Example

Topology of network encodes conditional independence assertions:



Weather is independent of the other variables

Toothache and *Catch* are conditionally independent given *Cavity*

Example

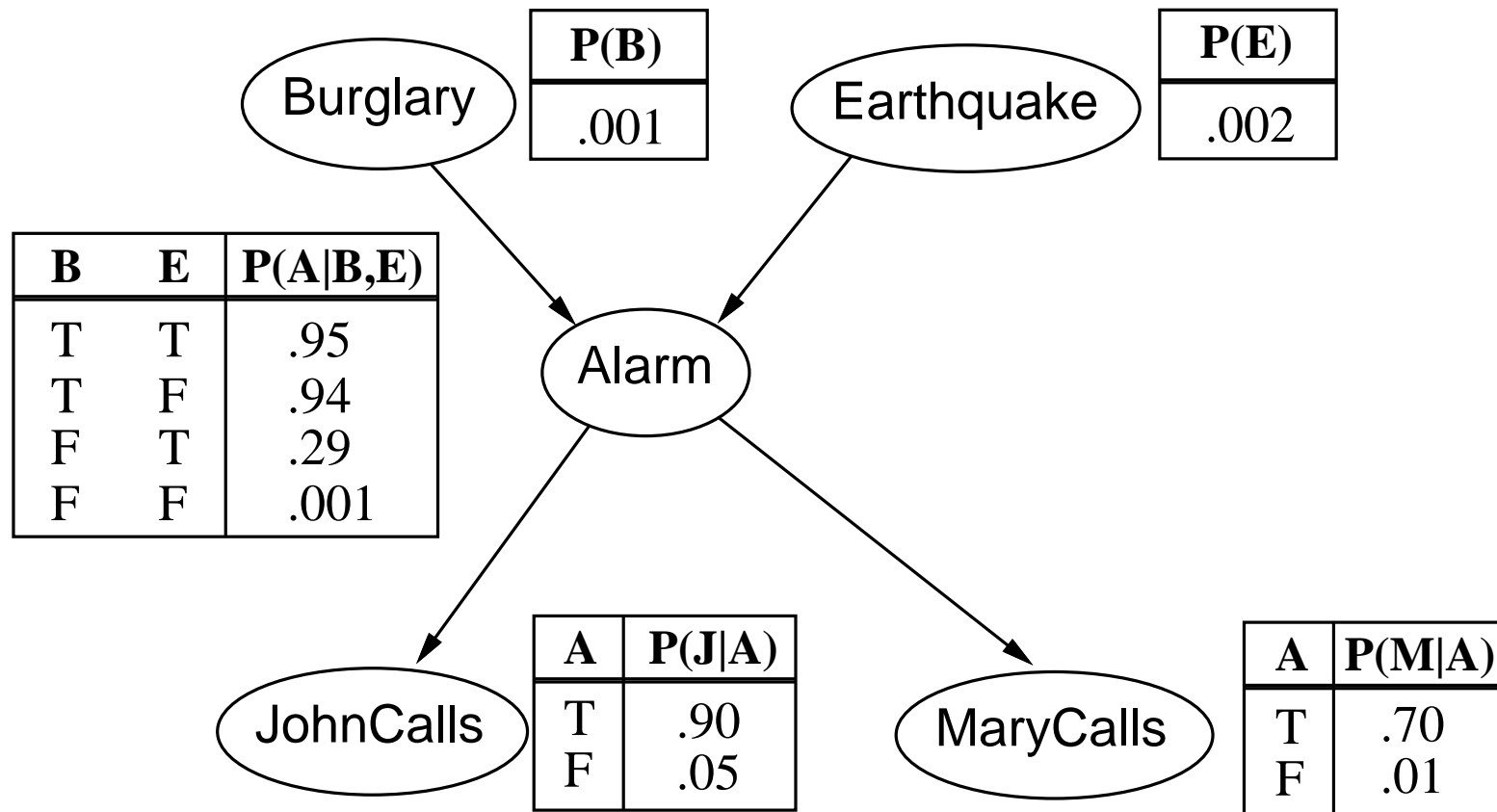
I'm at work, neighbor John calls to say my alarm is ringing, but neighbor Mary doesn't call. Sometimes it's set off by minor earthquakes. Is there a burglar?

Variables: *Burglar*, *Earthquake*, *Alarm*, *JohnCalls*, *MaryCalls*

Network topology reflects “causal” knowledge:

- A burglar can set the alarm off
- An earthquake can set the alarm off
- The alarm can cause Mary to call
- The alarm can cause John to call

Example contd.



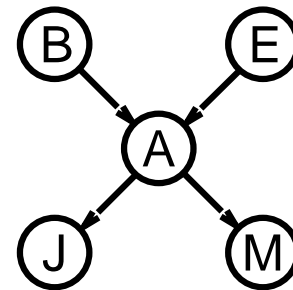
Global semantics

Global semantics defines the full joint distribution as the product of the local conditional distributions:

$$P(x_1, \dots, x_n) = \prod_{i=1}^n P(x_i | \text{parents}(X_i))$$

e.g., $P(j \wedge m \wedge a \wedge \neg b \wedge \neg e)$

=



Global semantics

“Global” semantics defines the full joint distribution as the product of the local conditional distributions:

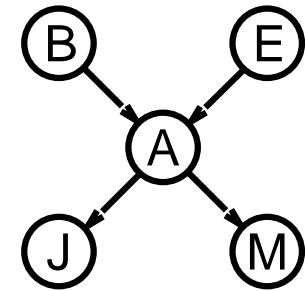
$$P(x_1, \dots, x_n) = \prod_{i=1}^n P(x_i | \text{parents}(X_i))$$

e.g., $P(j \wedge m \wedge a \wedge \neg b \wedge \neg e)$

$$= P(j|a)P(m|a)P(a|\neg b, \neg e)P(\neg b)P(\neg e)$$

$$= 0.9 \times 0.7 \times 0.001 \times 0.999 \times 0.998$$

$$\approx 0.00063$$

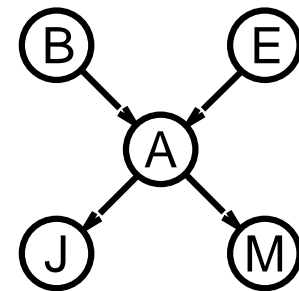


Inference by enumeration

Slightly intelligent way to sum out variables from the joint without actually constructing its explicit representation

Simple query on the burglary network:

$$\begin{aligned} & \mathbf{P}(B|j, m) \\ &= \mathbf{P}(B, j, m) / P(j, m) \\ &= \alpha \mathbf{P}(B, j, m) \\ &= \alpha \sum_e \sum_a \mathbf{P}(B, e, a, j, m) \end{aligned}$$



Rewrite full joint entries using product of CPT entries:

$$\begin{aligned} & \mathbf{P}(B|j, m) \\ &= \alpha \sum_e \sum_a \mathbf{P}(B)P(e)\mathbf{P}(a|B, e)P(j|a)P(m|a) \\ &= \alpha \mathbf{P}(B) \sum_e P(e) \sum_a \mathbf{P}(a|B, e)P(j|a)P(m|a) \end{aligned}$$

Recursive depth-first enumeration: $O(n)$ space, $O(d^n)$ time

Inference by stochastic simulation

Basic idea:

- 1) Draw N samples from a sampling distribution S
- 2) Compute an approximate posterior probability \hat{P}
- 3) Show this converges to the true probability P

0.5

Coin

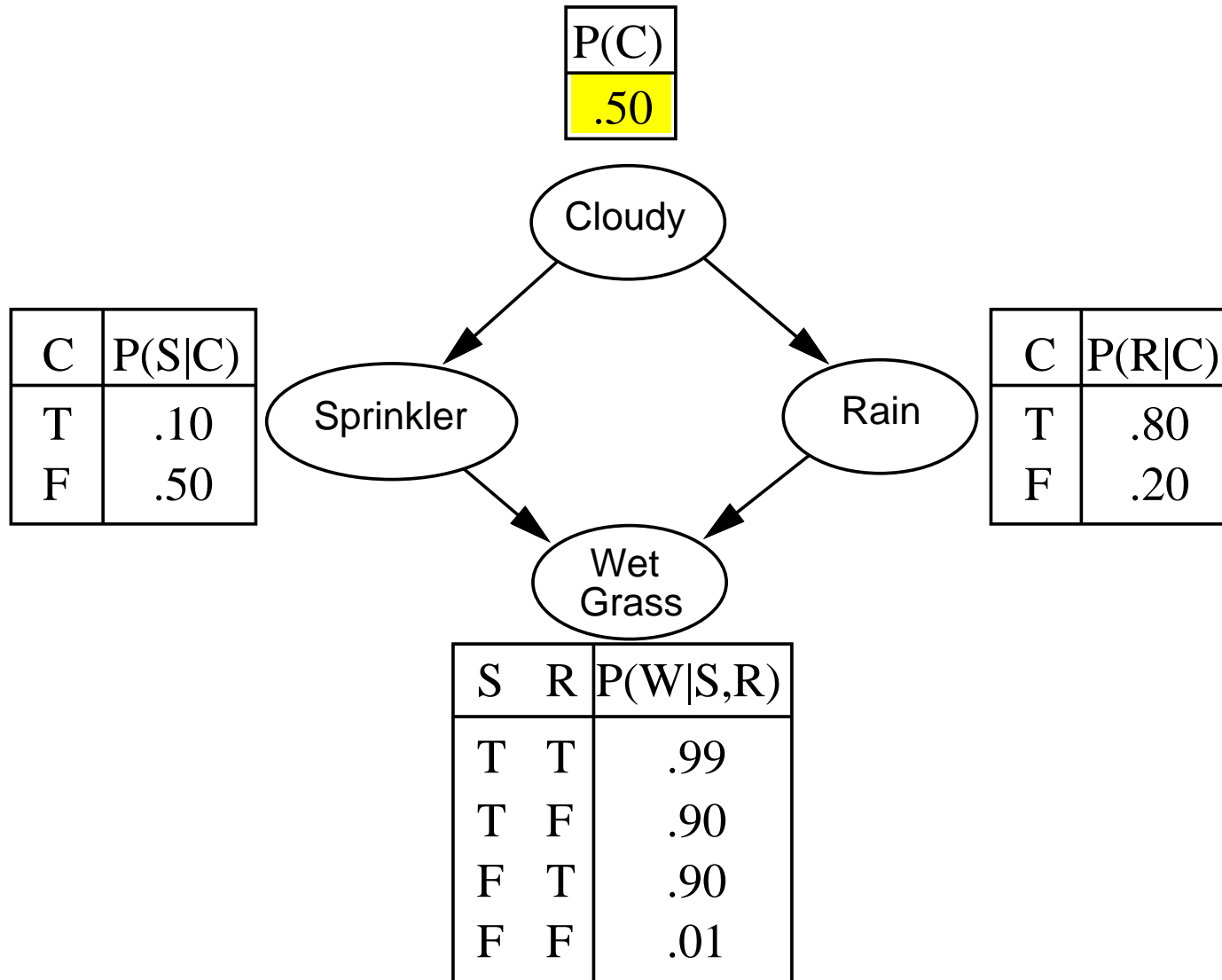
Outline:

- Sampling from an empty network
- Rejection sampling: reject samples disagreeing with evidence
- Likelihood weighting: use evidence to weight samples
- Markov chain Monte Carlo (MCMC): sample from a stochastic process whose stationary distribution is the true posterior

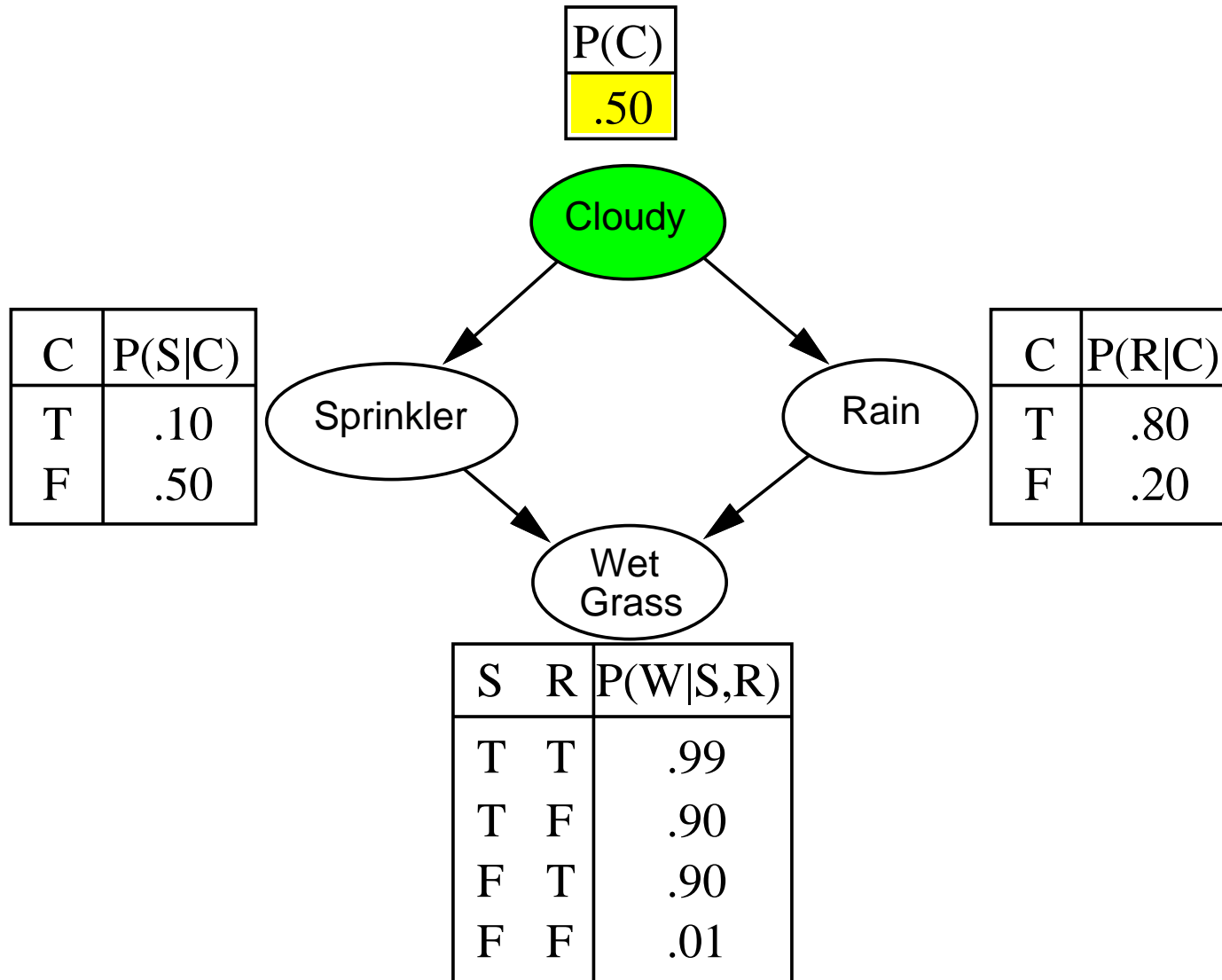
Sampling from an empty network

```
function PRIOR-SAMPLE( $bn$ ) returns an event sampled from  $bn$   
  inputs:  $bn$ , a belief network specifying joint distribution  $P(X_1, \dots, X_n)$   
   $\mathbf{x} \leftarrow$  an event with  $n$  elements  
  for  $i = 1$  to  $n$  do  
     $x_i \leftarrow$  a random sample from  $P(X_i \mid \text{parents}(X_i))$   
    given the values of  $\text{Parents}(X_i)$  in  $\mathbf{x}$   
  return  $\mathbf{x}$ 
```

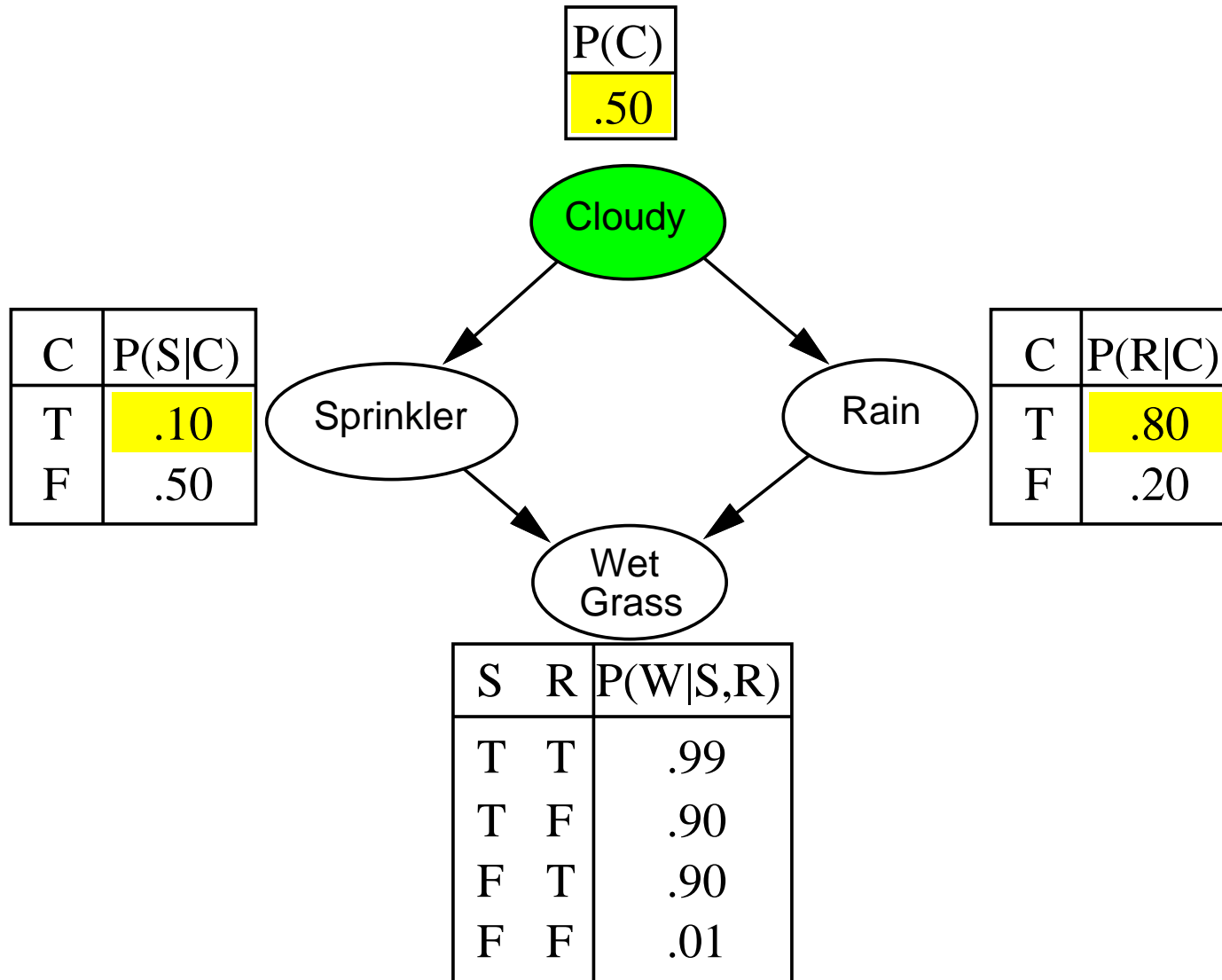
Example



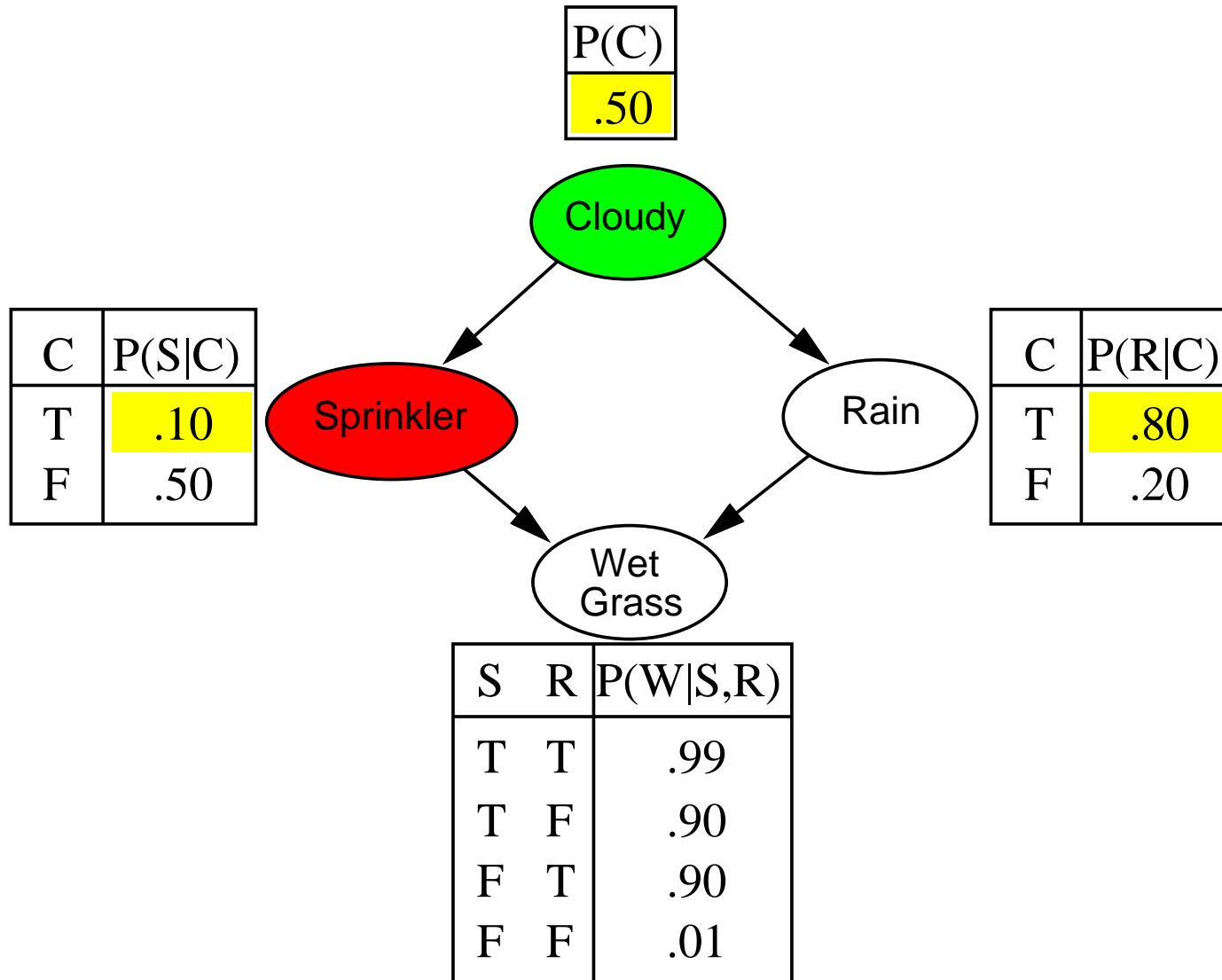
Example



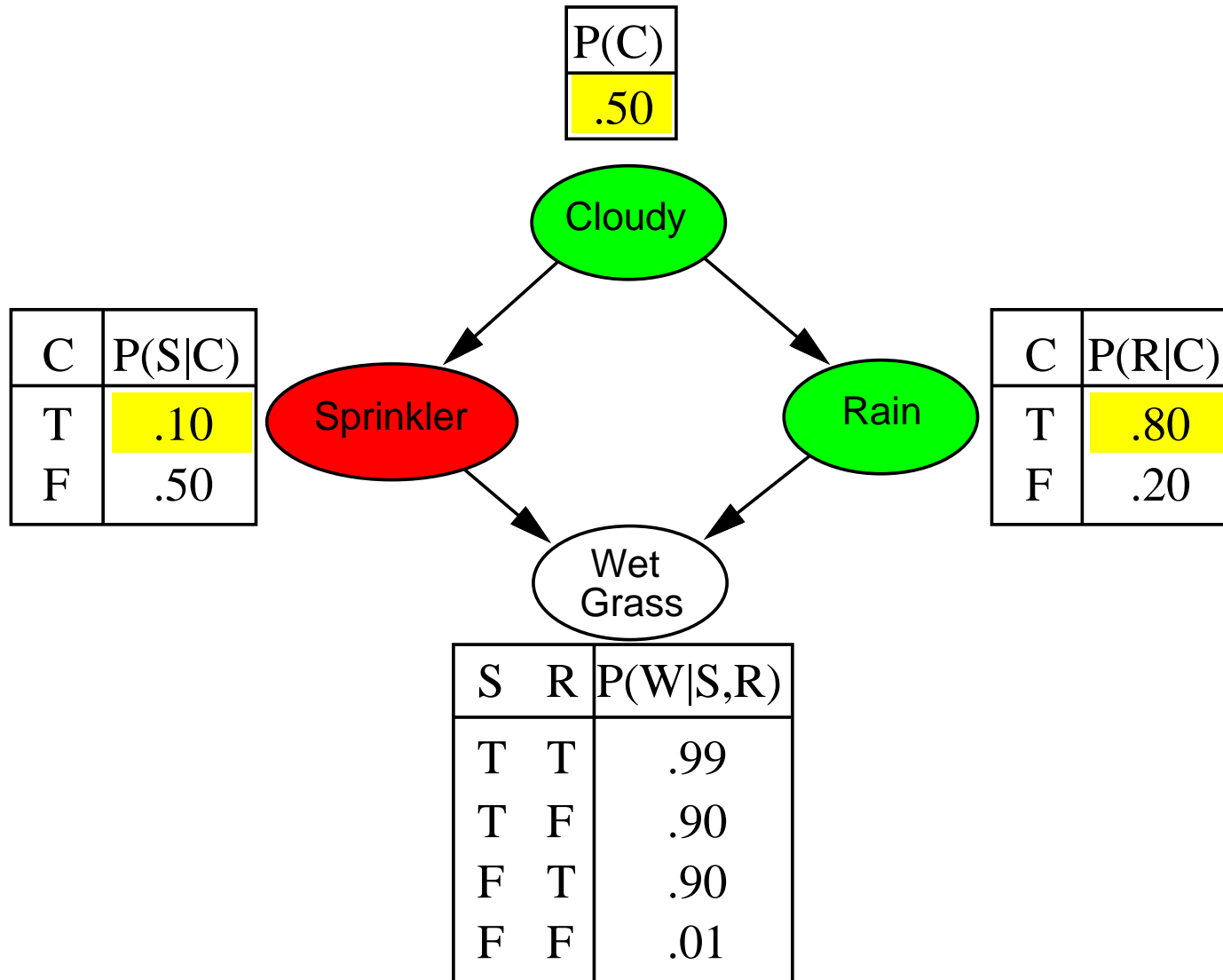
Example



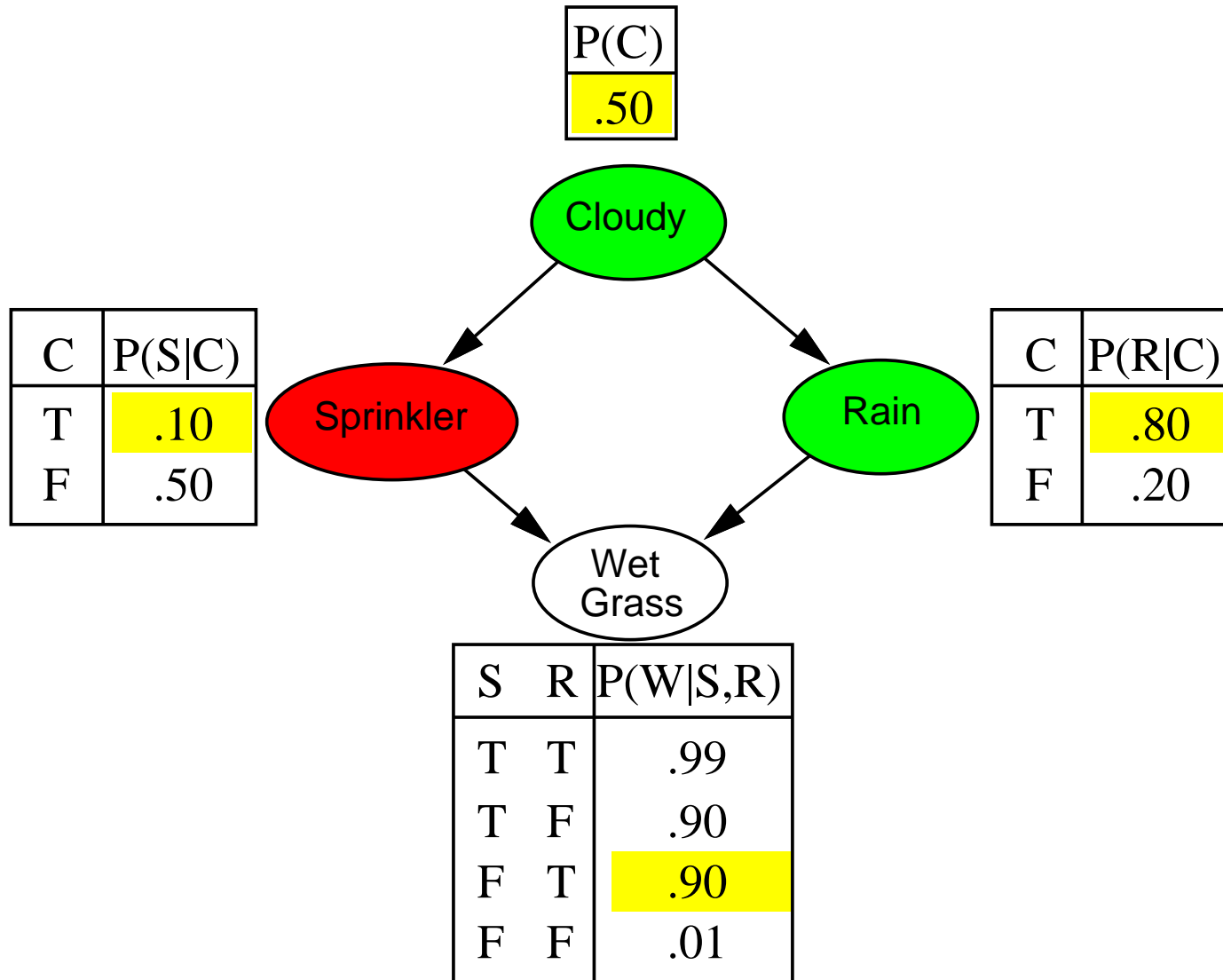
Example



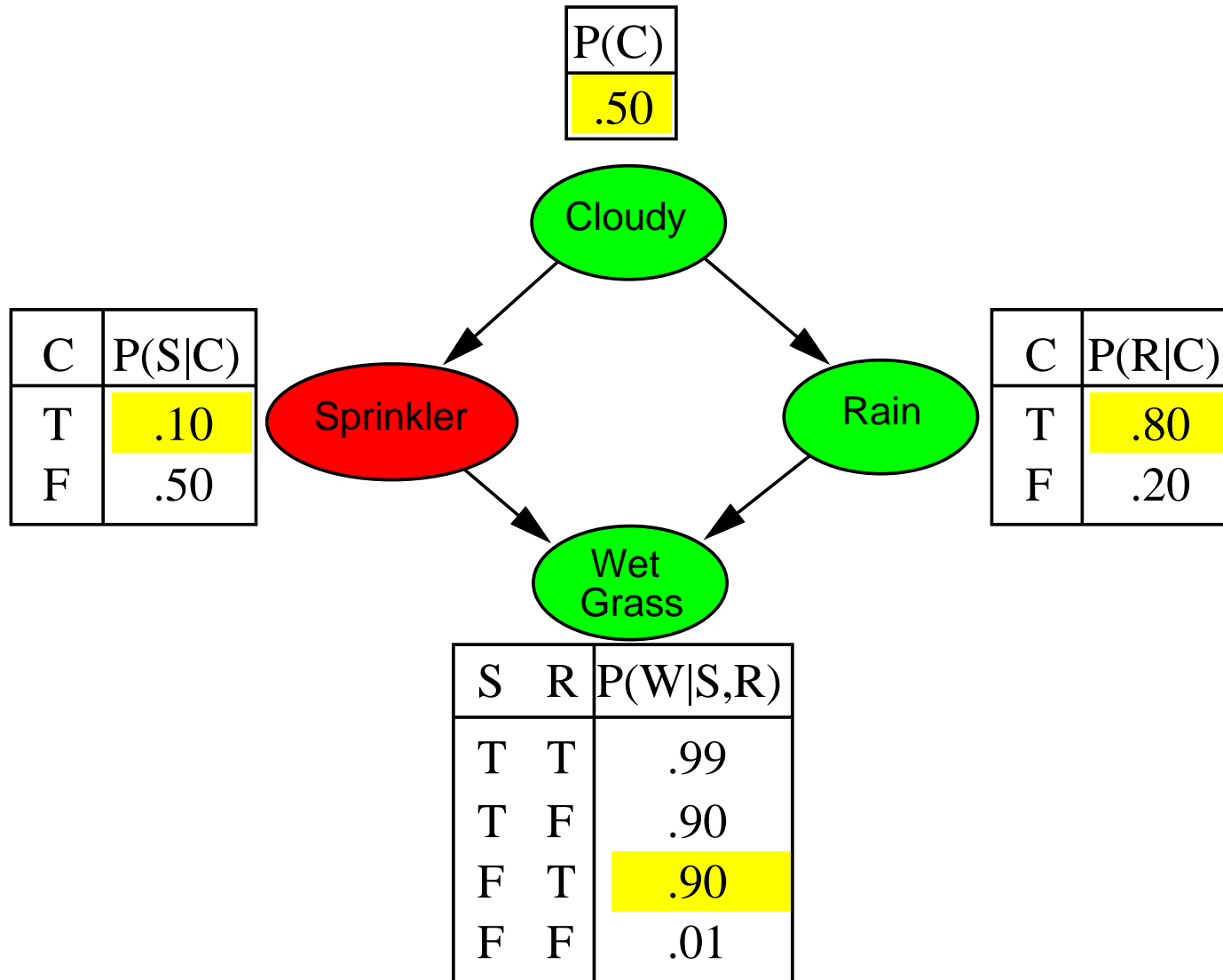
Example



Example



Example



Approximate inference using MCMC

“State” of network = current assignment to all variables.

Generate next state by sampling one variable given Markov blanket

Sample each variable in turn, keeping evidence fixed

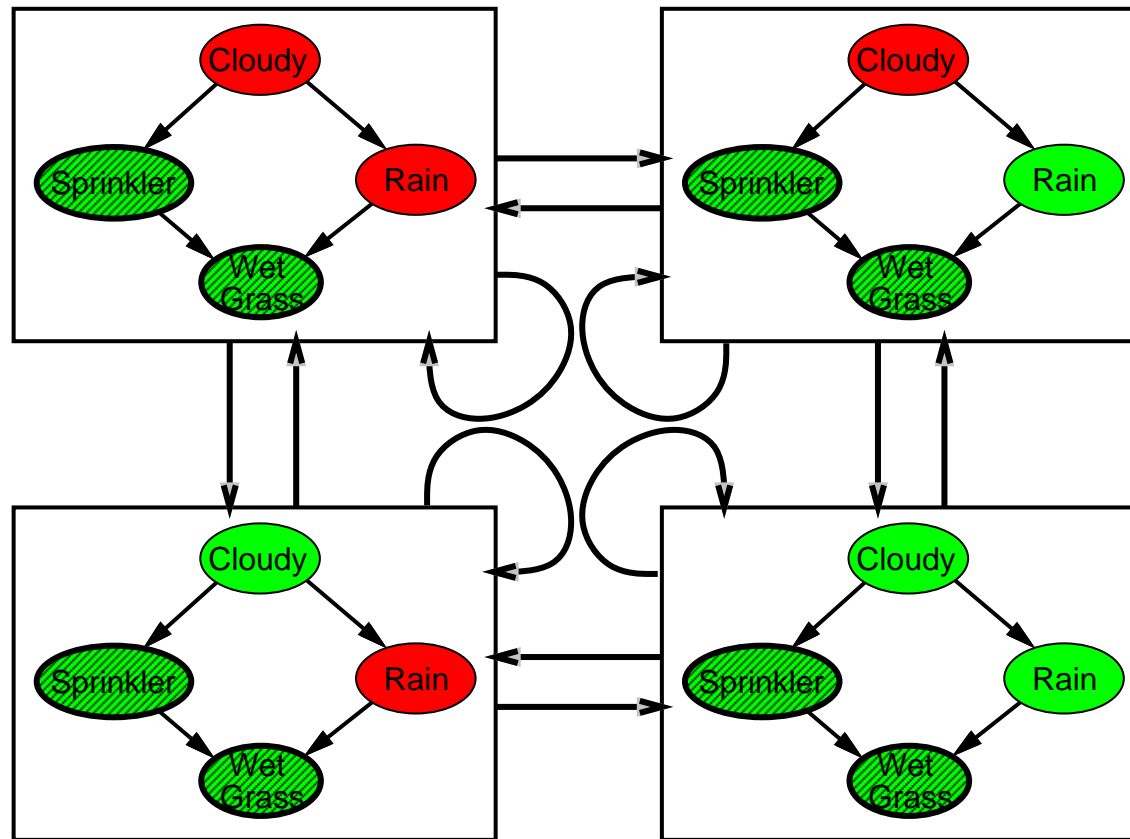
```
function MCMC-Ask( $X, \mathbf{e}, bn, N$ ) returns an estimate of  $P(X|\mathbf{e})$ 
  local variables:  $\mathbf{N}[X]$ , a vector of counts over  $X$ , initially zero
                   $\mathbf{Z}$ , the nonevidence variables in  $bn$ 
                   $\mathbf{x}$ , the current state of the network, initially copied from  $\mathbf{e}$ 

  initialize  $\mathbf{x}$  with random values for the variables in  $\mathbf{Y}$ 
  for  $j = 1$  to  $N$  do
    for each  $Z_i$  in  $\mathbf{Z}$  do
      sample the value of  $Z_i$  in  $\mathbf{x}$  from  $P(Z_i|mb(Z_i))$ 
        given the values of  $MB(Z_i)$  in  $\mathbf{x}$ 
       $\mathbf{N}[x] \leftarrow \mathbf{N}[x] + 1$  where  $x$  is the value of  $X$  in  $\mathbf{x}$ 
  return NORMALIZE( $\mathbf{N}[X]$ )
```

Can also choose a variable to sample at random each time

The Markov chain

With *Sprinkler* = true, *WetGrass* = true, there are four states:



Wander about for a while, average what you see

MCMC example contd.

Estimate $\mathbf{P}(Rain|Sprinkler = true, WetGrass = true)$

Sample *Cloudy* or *Rain* given its Markov blanket, repeat.
Count number of times *Rain* is true and false in the samples.

E.g., visit 100 states

31 have *Rain = true*, 69 have *Rain = false*

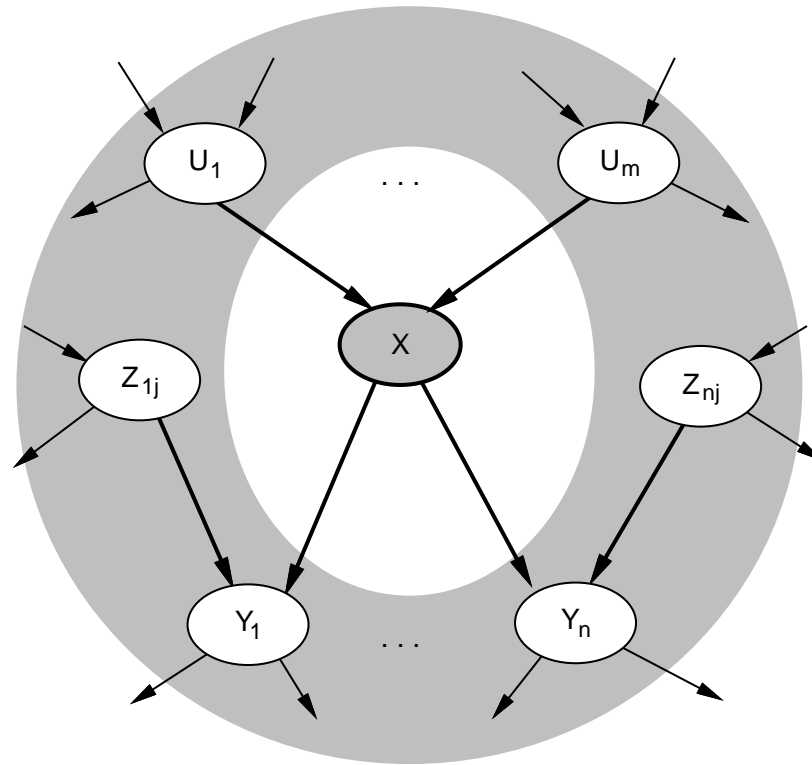
$$\begin{aligned}\hat{\mathbf{P}}(Rain|Sprinkler = true, WetGrass = true) \\ = \text{NORMALIZE}(\langle 31, 69 \rangle) = \langle 0.31, 0.69 \rangle\end{aligned}$$

Theorem: chain approaches **stationary distribution**:

long-run fraction of time spent in each state is exactly
proportional to its posterior probability

Markov blanket

Each node is conditionally independent of all others given its
Markov blanket: parents + children + children's parents



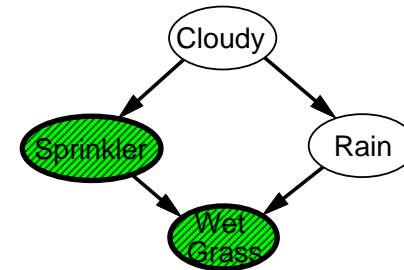
Markov blanket sampling

Markov blanket of *Cloudy* is

Sprinkler and *Rain*

Markov blanket of *Rain* is

Cloudy, *Sprinkler*, and *WetGrass*



Probability given the Markov blanket is calculated as follows:

$$P(x'_i | mb(X_i)) = P(x'_i | parents(X_i)) \prod_{Z_j \in Children(X_i)} P(z_j | parents(Z_j))$$

Easily implemented in message-passing parallel systems, brains

Main computational problems:

- 1) Difficult to tell if convergence has been achieved
- 2) Can be wasteful if Markov blanket is large:

$P(X_i | mb(X_i))$ won't change much (law of large numbers)

Time and uncertainty

The world changes; we need to track and predict it

Diabetes management vs vehicle diagnosis

Basic idea: copy state and evidence variables for each time step

\mathbf{X}_t = set of unobservable state variables at time t
e.g., *BloodSugar_t*, *StomachContents_t*, etc.

\mathbf{E}_t = set of observable evidence variables at time t
e.g., *MeasuredBloodSugar_t*, *PulseRate_t*, *FoodEaten_t*

This assumes **discrete time**; step size depends on problem

Notation: $\mathbf{X}_{a:b} = \mathbf{X}_a, \mathbf{X}_{a+1}, \dots, \mathbf{X}_{b-1}, \mathbf{X}_b$

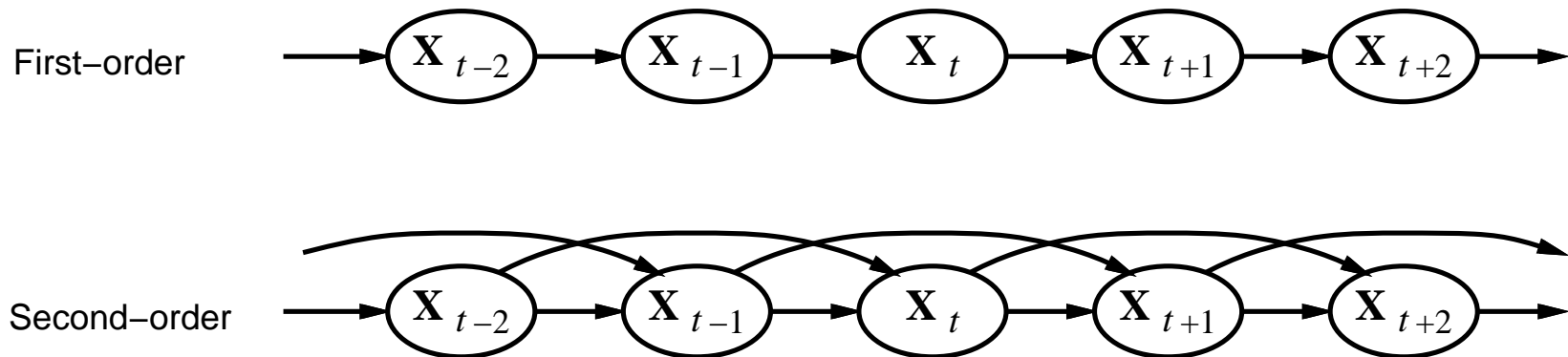
Markov processes (Markov chains)

Construct a Bayes net from these variables: parents?

Markov assumption: \mathbf{X}_t depends on **bounded** subset of $\mathbf{X}_{0:t-1}$

First-order Markov process: $\mathbf{P}(\mathbf{X}_t | \mathbf{X}_{0:t-1}) = \mathbf{P}(\mathbf{X}_t | \mathbf{X}_{t-1})$

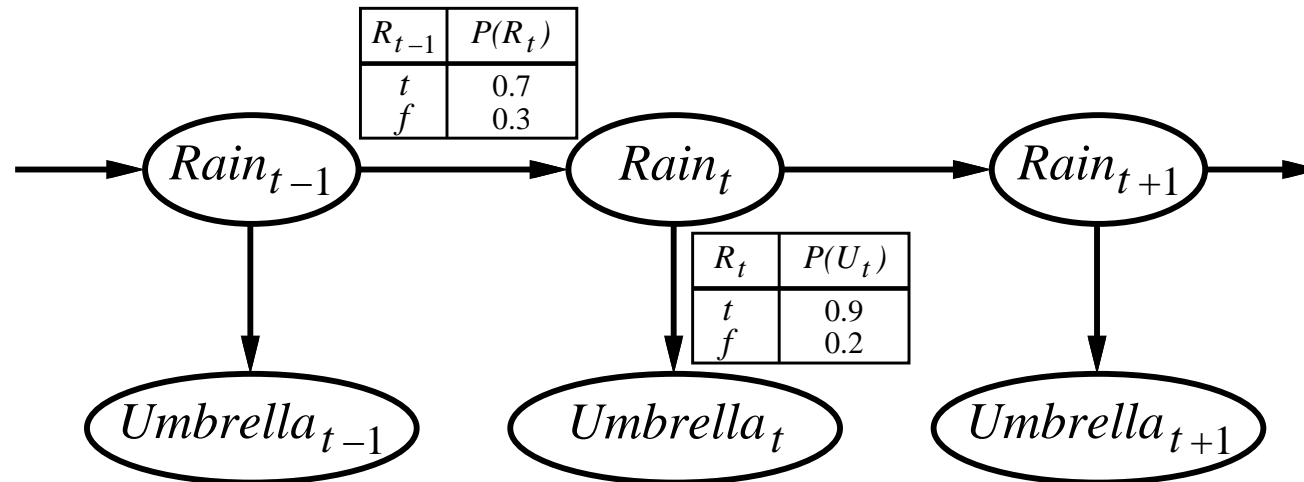
Second-order Markov process: $\mathbf{P}(\mathbf{X}_t | \mathbf{X}_{0:t-1}) = \mathbf{P}(\mathbf{X}_t | \mathbf{X}_{t-2}, \mathbf{X}_{t-1})$



Sensor Markov assumption: $\mathbf{P}(\mathbf{E}_t | \mathbf{X}_{0:t}, \mathbf{E}_{0:t-1}) = \mathbf{P}(\mathbf{E}_t | \mathbf{X}_t)$

Stationary process: transition model $\mathbf{P}(\mathbf{X}_t | \mathbf{X}_{t-1})$ and sensor model $\mathbf{P}(\mathbf{E}_t | \mathbf{X}_t)$ fixed for all t

Example



First-order Markov assumption not exactly true in real world!

Possible fixes:

1. **Increase order** of Markov process
2. **Augment state**, e.g., add $Temp_t$, $Pressure_t$

Example: robot motion.

Augment position and velocity with $Battery_t$

Inference tasks

Filtering: $P(\mathbf{X}_t | \mathbf{e}_{1:t})$

belief state—input to the decision process of a rational agent

Prediction: $P(\mathbf{X}_{t+k} | \mathbf{e}_{1:t})$ for $k > 0$

evaluation of possible action sequences;

like filtering without the evidence

Smoothing: $P(\mathbf{X}_k | \mathbf{e}_{1:t})$ for $0 \leq k < t$

better estimate of past states, essential for learning

Most likely explanation: $\arg \max_{\mathbf{x}_{1:t}} P(\mathbf{x}_{1:t} | \mathbf{e}_{1:t})$

speech recognition, decoding with a noisy channel

Filtering

Aim: devise a **recursive** state estimation algorithm:

$$\mathbf{P}(\mathbf{X}_{t+1}|\mathbf{e}_{1:t+1}) = f(\mathbf{e}_{t+1}, \mathbf{P}(\mathbf{X}_t|\mathbf{e}_{1:t}))$$

$$\begin{aligned}\mathbf{P}(\mathbf{X}_{t+1}|\mathbf{e}_{1:t+1}) &= \mathbf{P}(\mathbf{X}_{t+1}|\mathbf{e}_{1:t}, \mathbf{e}_{t+1}) \\ &= \alpha \mathbf{P}(\mathbf{e}_{t+1}|\mathbf{X}_{t+1}, \mathbf{e}_{1:t}) \mathbf{P}(\mathbf{X}_{t+1}|\mathbf{e}_{1:t}) \\ &= \alpha \mathbf{P}(\mathbf{e}_{t+1}|\mathbf{X}_{t+1}) \mathbf{P}(\mathbf{X}_{t+1}|\mathbf{e}_{1:t})\end{aligned}$$

I.e., **prediction** + **estimation**. Prediction by summing out \mathbf{X}_t :

$$\begin{aligned}\mathbf{P}(\mathbf{X}_{t+1}|\mathbf{e}_{1:t+1}) &= \alpha \mathbf{P}(\mathbf{e}_{t+1}|\mathbf{X}_{t+1}) \sum_{\mathbf{x}_t} \mathbf{P}(\mathbf{X}_{t+1}|\mathbf{x}_t, \mathbf{e}_{1:t}) P(\mathbf{x}_t|\mathbf{e}_{1:t}) \\ &= \alpha \mathbf{P}(\mathbf{e}_{t+1}|\mathbf{X}_{t+1}) \sum_{\mathbf{x}_t} \mathbf{P}(\mathbf{X}_{t+1}|\mathbf{x}_t) P(\mathbf{x}_t|\mathbf{e}_{1:t})\end{aligned}$$

$\mathbf{f}_{1:t+1} = \text{FORWARD}(\mathbf{f}_{1:t}, \mathbf{e}_{t+1})$ where $\mathbf{f}_{1:t} = \mathbf{P}(\mathbf{X}_t|\mathbf{e}_{1:t})$

Time and space **constant** (independent of t)

Filtering example

